1분반 20151002학번 컴퓨터학과 서유정

Vladimir Quicksort 코드구현(java)

**import** java.util.Random;

**public** **class** Vladimir {

**public** **static** **int** *p1*;

**public** **static** **int** *p2*;

**public** **static** **int** *array\_size* = 800000;

**public** **static** **int** *L* = 129;

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

//배열을 랜덤하게 생성

Random r = **new** Random();

**int**[] arr = **new** **int**[*array\_size*];

**for**(**int** i = 0; i< arr.length; i++)

arr[i] = r.nextInt(100000);

//vladimir 수행하고 실행시간 check

**long** beginTime = System.*currentTimeMillis*();

*VladimirSort*(arr, 0, (arr.length-1));

**long** endTime = System.*currentTimeMillis*();

**long** ms = endTime - beginTime;

**float** sec = (**float**) ms / 1000;

**float** min = (**float**) sec / 60;

**if** (min > 1) {

System.***out***.println("Vladimir수행시간: " + min + "(min)");

} **else** **if** (sec > 1) {

System.***out***.println("Vladimir수행시간: " + sec + "(sec)");

} **else** {

System.***out***.println("Vladimir수행시간: " + ms + "(ms)");

}

}

**public** **static** **void** swap (**int**[] arr, **int** a, **int** b) {

**int** temp = arr[a];

arr[a] = arr[b];

arr[b] = temp;

}

//L값보다 적으면 insertionsort 적용하고, 크면 dualpivotquicksort 적용

**public** **static** **void** VladimirSort(**int**[] arr, **int** l, **int** r) {

**if**(l < r) {

**if**((r-l) < *L*)

*InsertionSort*(arr, l , r);

**else** {

*Partition*(arr, l, r);

*VladimirSort*(arr, l, *p1* -1);

*VladimirSort*(arr, *p2* + 1, r);

*VladimirSort*(arr, *p1* + 1, *p2* - 1);

}

}

}

**public** **static** **void** InsertionSort(**int**[] arr, **int** l, **int** r) {

**for**(**int** i = l + 1; i <= r; i++) {

**int** loc = i - 1;

**int** key = arr[i];

**while**(loc >= l && key < arr[loc]) {

arr[loc + 1] = arr[loc];

loc--;

}

arr[loc + 1] = key;

}

}

**public** **static** **void** Partition(**int**[] arr, **int** l, **int** r) {

**if**(arr[l] > arr[r])

*swap*(arr,l,r);

**int** pivot1 = arr[l], pivot2 = arr[r];

**int** cc = l + 1; //현재 체크중인 index

**int** lc= l + 1;//pivot1보다 작은값 index

**int** rc= r - 1; //pivot2보다 큰 index

**while**(lc <= rc) {

**if**(arr[lc] < pivot1) {

*swap*(arr, lc, cc);

cc++;

}

**else** **if** (arr[lc] >= pivot2) {

**while** ((arr[rc] > pivot2) && (lc < rc))

rc--;

*swap*(arr, lc, rc);

rc--;

**if**(arr[lc] < pivot1) {

*swap*(arr, lc, cc);

cc++;

}

}

lc++;

}

cc--;

rc++;

*swap*(arr, l, cc);

*swap*(arr, r, rc);

*p1* = cc;

*p2* = rc;

}

}